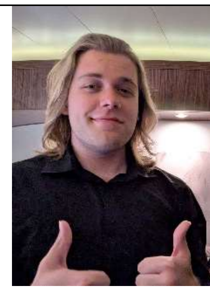
SP-104 Active Learning System Images

Software Requirements Specification (SRS)

CS 4850 – Section 02 – Fall 2025

September 21, 2025

Matthew Hall

			
Josh Smith Team Leader Developer	Elijah Merrill Developer	Noah Lane Documentation	Matthew Hall Documentation

Team Members:

Name	Role	Cell / Alt Email
Josh Smith (Team Lead)	Developer	706-414-2827 joshuasmith0515@gmail.com
Elijah Merrill	Developer	478-283-0811 elijahmerrill04@gmail.com
Matthew Hall	Documentation	678-873-8542 thematthewhall7@gmail.com
Noah Lane	Documentation	706-591-2312 noahlane142@gmail.com
Sharon Perry	Advisor	770-329-3895 sperry46@kennesaw.edu

1.0 Introduction *3*
1.1 Overview.......... 3
1.2 Project Scope.......... 3
1.3 Definitions and Acronyms 3
1.4 Assumptions.......... 4
1.5 References 3
2.0 Constraints *5*
2.1 Environment 5
2.2 System 5
3.0 Functional Requirements.......... *5*
3.1 Dataset Loading 5
3.2 Initial Model Training 6
3.3 Active Learning Loop.......... 7
3.4 Performance Evaluation 7
3.5 User Interface 8
4.0 Non-Functional Requirements *9*
4.1 Maintainability 9
4.2 Usability 8
5.0 External Interface Requirements *9*
5.1 User Interface Requirements 9
5.2 Hardware Interface Requirements.......... 9
5.3 Software Interface Requirements 9
6.0 Analysis *10*
6.1 Use Cases..........10
6.2 Data Flow..........10

1.0 Introduction

1.1 Overview

This document outlines the requirements for an Active Learning System using medical images of Chest X-Rays from the NIH Chest X-Rays dataset via Kaggle. The goal of the system is to reduce the manual effort needed for image annotation by incorporating an active learning loop. Through uncertainty sampling and iterative model training, the system will prioritize the most informative images for labeling, resulting in improved efficiency and accuracy.

1.2 Project Scope

The scope of this project is to design and implement an Active Learning System for labeling medical images using the NIH Chest X-Rays dataset. The system will focus on reducing the manual annotation effort by employing active learning techniques, specifically uncertainty sampling, to select the most informative images for labeling. Core functionality will include loading the dataset, training a model with a set limit of initial labels, implementing an iterative active learning loop, and evaluating the performance through metrics and learning curve analysis. The system will be delivered as either a web-based tool, an interactive notebook, or a full-on downloadable application.

This project does not aim to function as a diagnostic tool for clinical use, nor will it provide integration with hospital database record systems. The primary focus is to create a functional prototype that demonstrates the benefits of active learning with medical image classification, while clearly defining the boundaries of what the system will and will not deliver.

1.3 Definitions and Acronyms

Active Learning – Machine learning approach where the model actively selects unlabeled samples to be labeled, reducing the overall labeling effort while improving training efficiency.

Uncertainty Sampling – A Strategy used with active learning where the system queries samples that the model is least confident in with predictions. Samples will then be prioritized for labeling.

NIH – Full Name: National Library of Medicine. An organization where many articles for this document were taken.

Metadata – Supplementary information that describes data attributes. When it comes to the NIH dataset this would include disease labels and image file paths.

IDEAL – Full name: Interpretation-based Deep Active Learning.

AUC-ROC (Area Under the Receiver Operating Characteristic Curve) – A performance metric for classification that measures trade-offs between true positive rates and false positive rates.

AUC – PR (Area Under the Precision-Recall Curve) – A performance metric mainly used for imbalanced datasets. Summarizes the trade-offs between precision and recall.

1.4 Assumptions

1. Assumed that the NIH Chest X-Rays dataset from Kaggle will remain publicly available and accessible for the duration of development
2. Assumed that users of the system will have access to a machine capable of running the program as a web-based tool, interactive notebook, or downloadable application.
3. Assumed that users will have access to an internet connection for downloading the dataset and installing any external dependencies.
4. Assumed that the hosting environment (given if it's deployed as a web-based tool) will support the necessary frameworks and allow secure storage of temporary data.

1.5 References

- X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, and R. M. Summers, “ChestX-Ray14: Data,” *Kaggl*e, 2017. Available: https://www.kaggle.com/datasets/nih-chest-xrays/data
- G. M. Roy, M. Zandehshahvar, M. van Assen, N. Kittisut, K. Peters, C. N. De Cecco and A. Adibi, “Deep Active Learning for Lung Disease Severity Classification from Chest X-rays: Learning with Less Data in the Presence of Class Imbalance,” *arXiv* August 2025. Available: https://arxiv.org/abs/2508.21263
- D. Mahapatra, “Interpretability-Driven Sample Selection Using Self Supervised Learning For Disease Classification And Segmentation,” *arXiv* Apr. 2021. Available: https://arxiv.org/abs/2104.06087
- H. Wang, Q. Jin, S. Li, S. Wang, S. Liu, M. Wang, Z. Song, “A Comprehensive Survey on Deep Active Learning in Medical Image Analysis,” *arXiv* Oct. 2023. Available: https://arxiv.org/abs/2310.14230
- A. Giełczyk, A. Marciniak, M. Tarczewska, Z. Lutowski, “Pre-processing methods in chest X-ray image classification”, *PubMed Central* Apr. 2017. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC8982897/

2.0 Constraints

This section details constraints that the environment of the system and the system itself will likely face. While some of these constraints will not apply to some users, others may face issues pertaining to the constraints mentioned below.

2.1 Environment

The host environment does not require higher-end internal components. However, the host GPU and CPU should be at least somewhat adequate (e.g. modern processor/graphics cards). While all operating systems will be allowed, the system will only run on the more recent systems (e.g. Windows 10/11, macOS Sequoia/Ventura/Tahoe, Linux). No network bandwidth requirements as the system will likely only be for general use and not for use within a hospital/clinic. Furthermore, training of the model will be done locally.

2.2 System

It must be built in a timeframe of approximately 2 months. The system should not be constrained to any specific dataset, as being able to use any chest X-ray dataset with it makes it more usable. The system may be constrained by the processing power of the imported libraries used for developing the learning model. Additional constraints may include the effect of the very large datasets that will be used for testing the system (e.g., 40-520 GB). While not super important to the function of the system, the system will be held back by more niche data that is not widely available to the public due to HIPPA laws and a lack of incentives for large medical institutions.

3.0 Functional Requirements

3.1 Dataset Loading

The system should be capable of loading and preprocessing the NIH Chest X-ray dataset to ensure it is prepared for training and active learning tasks. The dataset contains over 112,000 chest X-ray images accompanied by metadata like disease labels. The dataset loading process must therefore involve both image files and metadata in order to maintain consistency and allow for accurate filtering and labeling.

During this stage, the system will filter the images into various view types (TBD) and address issues of data integrity when it comes to handling corrupted images. The metadata must also be carefully parsed so that labels with missing, ambiguous, or no findings are either handled appropriately or flagged for exclusion. By enforcing these checks, the dataset loading module ensures that the input data remains reliable for training and evaluation purposes.

Once the dataset has been validated, images must be standardized into a consistent format that works with the machine learning model. This would include resizing images to a fixed resolution, such as 224x224 or 512x512 pixels, while maintaining aspect ratios or applying padding if necessary. Since chest X-rays are greyscale, intensity normalization must also be applied to ensure uniform pixel value distributions across the dataset. In some cases, additional preprocessing techniques such as contrast enhancement or histogram equalization may be used to improve image clarity. These steps help ensure any variations on images due to their quality, size, or format so it does not negatively impact model performance.

The dataset loading process must also partition the data into subsets, specifically training, validation, and test subsets, with the use of reproducible methods. Proper class distribution should be set in place to help avoid any bias that could occur. Finally, data storage and access should have a consistent folder structure, organizing images alongside their labels, ensuring efficient data retrieval.

In summary, the dataset loading functionally establishes all subsequent stages for the active learning system. By ensuring images and the metadata go through this preprocessing system, the system provides inputs for baseline training and performance evaluation. These steps are essential not only for system accuracy but also for ensuring that results are reproducible.

3.2 Initial Model Training

The system should train a baseline model using a limited, labeled subset of the chest X-ray data to establish a performance benchmark before active learning begins. Within the initial model training phase, transfer learning should be leveraged, using datasets pre-trained on large image datasets and fine-tuned on chest X-ray tasks (TBD). The reason for this is mainly due to the class imbalance (some diseases being rarer than others) in the NIH Chest X-ray dataset. This could lead to issues, given that training the model from scratch means the model must learn low-level image features such as edges and texture from chest X-rays. Granted, this is feasible, but from research, it could be computationally expensive. Training via transfer learning (pretrained models) helps reduce training time, improve convergence stability, and possibly even achieve higher performance with fewer epochs. For example, previous work (such as Anatomy-XNet) used a dataset called DenseNet-121 that was pretrained on ImageNet as the backbone, then adding attention and pooling modules to improve the detection of thoracic diseases.

Training must be conducted using standard procedures such as splitting the labeled subset into training and validation subsets with the use of an optimizer like AdamW, choosing appropriate loss functions suited for imbalanced classes, and monitoring performance with metrics. The system should also support data augmentation to increase the effective size of the labeled set and should use reproducible settings so the results can be compared consistently.

Finally, the initial model training is not only to produce a baseline, but also to verify that preprocessing and dataset loading steps are working correctly, making sure label handling is appropriate and that the class label imbalance or data quality does not become overwhelmed, or the model begins to have bias. If baseline performance is very low, this phase may help expose issues to fix before entering the active learning loop.

3.3 Active Learning Loop

The system should implement an active learning loop to intelligently select which unlabeled chest X-ray images to label next, thereby reducing the annotation burden while maintaining or improving model performance. After training the baseline model, the system should maintain two pools of data: a labeled pool and an unlabeled pool. In each iteration of the loop, the model will evaluate the unlabeled pool to determine which samples are most informative for improving the model. These selected samples will then be labeled using annotations from the chest x-ray dataset (simulating an expert oracle), added to the labeled pool, and removed from the unlabeled pool. The model will then be retrained using the expanded labeled data. This loop repeats until a stopping criterion is met, which could be some form of labeling budget or exhausting the unlabeled pool.

Key to the loop is the acquisition strategy, which helps determine information on samples. Common strategies would include uncertainty sampling, where the model would choose samples, it is least confident about. In the context of chest X-rays, studies like IDEAL have shown that combining uncertainty-based selection with other signals can lead to better sample selection when compared to purely random or naïve methods. Furthermore, recent work in *Deep Active Learning for Lung Disease Severity Classification from Chest X-rays* demonstrates how uncertainty sampling (using Monte Carlo Dropout and other Bayesian approximations) plus weighted loss functions can help address class imbalance and reduce labeling needs significantly.

Finally, the loop must include mechanisms for performance measurements after each iteration. Metrics should be computed not only on training/validation subsets but also on test sets to monitor generalization. Learning curves vs the number of labeled samples gives insight into how effectively the active learning strategy is working and helps decide when to stop.

3.4 Performance Evaluation

The system should include a performance evaluation to measure how well the model improves over time under an active learning regime, ensuring the gains in labeling efficiency do not come at the cost of diagnostic accuracy or fairness. Performance should be assessed using a set of standard classification metrics computed at each iteration of the active learning loop. Performance metrics should include accuracy, precision, recall, F1-score, Area Under the Receiver Operating Characteristic curve (AUC-ROC), and possibly Area Under the Precision-Recall Curve (AUC-PR). These metrics will capture different aspects of model behavior, such as how often it is correct overall and how confident predictions are ranked.

Evaluation must pay great attention to metrics that are sensitive to imbalance. For example, F1-scores balance precision and recall, and AUC-PR is more informative than AUC-ROC when dealing with highly skewed classes. Studies like *Deep Active Learning for Lung Disease Severity Classification from Chest X-rays* used both AUC-ROC and AUC-PR to assess performance as data labeling increased, showing that certain acquisition strategies can have a high AUC while using a fraction of labeled data.

In addition to the quantitative metrics, the system should produce learning curves that plot the performance against the number of labeled instances. This enables visualization of model improvement as more annotations are acquired and helps identify when marginal gains are declining. The evaluation should also involve cross-validation to ensure generalization beyond the immediate validation splits.

Finally, where feasible, evaluation should include statistical significance testing of differences between acquisition strategies or model versions (e.g., uncertainty sampling vs random sampling), confidence intervals for metrics like AUC (95% CI), and error analysis to investigate failure cases.

3.5 User Interface

The system should be accessible either through a web-based tool, an interactive notebook, or a downloadable application.

The system's user interface should provide a clean and efficient means for users to interact with the Active Learning System, performing tasks such as dataset loading, model training, sample selection, and evaluation. The interface should expose some controls for uploading or selecting the dataset, specifically, preprocessing options, initiating baseline model training, and configuring the active learning loop parameter. It should also present visualization components such as sample images and confidence scores to help users understand which images the model is uncertain about and why those images were chosen for labeling.

In addition to these elements, the interface must support responsive feedback mechanisms: showing progress for training and loading steps, error alerts, and status indicators so users always know what stage the system is in.

Lastly, the interface should allow for results to be visualized: learning curves (performance metrics vs. number of labeled samples) and possibly side-by-side comparison of model versions

(TBD). It should also enable the export of results in standard formats (e.g., PDF, PNG) so users can analyze data outside the tool.

4.0 Non-Functional Requirements

4.1 Maintainability

The system's codebase should be modular and well-documented to support future development. Components such as preprocessing, model training, evaluation, and the active learning loop should be designed for easy modification without affecting the whole system.

4.2 Usability

The system should provide a simple, intuitive interface to display data and metrics. Navigation should require minimal technical expertise with consistent terminology.

5.0 External Interface Requirements

5.1 User Interface Requirements

The system should provide a dashboard or interactive notebook interface that allows users to load datasets, start training, and view results. Performance metrics and learning curves should also be displayed in a clear, visual format.

5.2 Hardware Interface Requirements

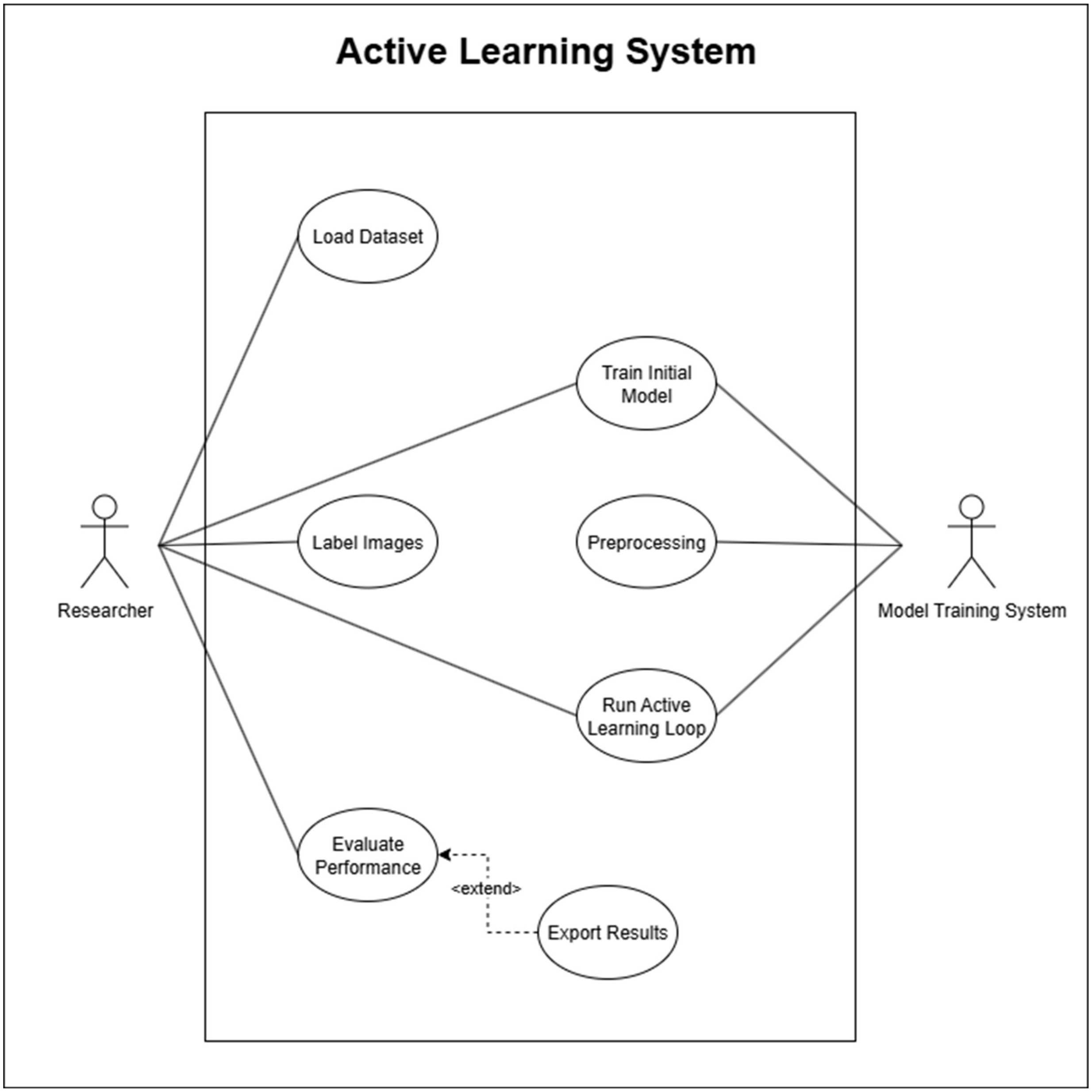
The system should be capable of running on a standard GPU-enabled workstation. The system should also not require specialized medical imaging hardware.

5.3 Software Interface Requirements

The system should integrate with Python-based deep learning frameworks like TensorFlow or PyTorch for model training. The system should also support importing the NIH Chest X-Ray dataset in common formats.

6.0 Analysis

6.1 Use Cases



6.2 Data Flow

The researcher has full access to the active learning system. They can load datasets, label images, and evaluate the model’s performance. The model training component manages preprocessing, training, and executes the active learning loops in coordination with the researcher. When evaluating results, the researcher also has the option to export performance metrics for further analysis.